Linux 101 for network administrators

Devdas Bhagat
<devdas.b@gmail.com>

A little history

- Linux is a Unix clone
  - Technically, Linux is the kernel. The userland is GNU.
- Started by a Finnish programming student named Linux Torvalds in 1991
- Licensed under the GPLv2

Unix design philosophy

- Small is beautiful
- Make each program do one thing well
  - Do not make your code do too many things
  - Factor into smaller units
- Prototype early, release often
  - Real world feedback is really important
- Portability is more important than efficiency
  - Computers are fast enough

Unix Design Philosophy

- Configuration lives in flat files
- Avoid captive user interfaces
- Make programs into filters

Unix design

- Kernel
  - Minimum OS
  - Provides abstractions over hardware and software
- Userspace
  - Everything else

Basic interaction

- The Unix shell
  - What people generally think of as “Unix”
  - Fundamental interface to any Unix system
- Multiple options:
  - /bin/sh is a standard location for a POSIX shell
  - bash, csh, dash, ksh, zsh are commonly used shells

User basics

- Two categories of users:
  - root
    - root is the prime administrative user account
    - Normal usage MUST NEVER be done as root
    - The root user prompt is usually a # for POSIX shells
    - root is **ALWAYS** uid 0.
    - All users with uid 0 have root privileges.
  - Everyone else
    - This is what you use for daily activities
    - Normal users get $ as a prompt for their shell

Getting around

- Starting off
  - $HOME
  - ls
  - cd
  - mkdir
  - touch
  - rmdir
  - rm
  - pwd

Help!

- man
- info
  - pinfo
- whatis
- whereis
- which
- The Internet!
  - Your favorite search engine!
  - IRC
    - irc.freenode.net and irc.oftc.net are fantastic

System status

- What's happening
  - w
  - who
  - uptime
  - last
  - top
  - ps

Files

- Everything is a file
  - Mostly
- Files are contained in filesystems.
  - Filesystems are an abstraction/organisation layer on raw storage
- Files are organised in a tree structure
- The directory separator is '/'
- '/' is also used for the root of the tree

Filesystems

- Data structures used to track files on disk
- Common fs types
- mkfs
- Key concepts
  - Superblock
  - inode
  - Data block
  - Directory block
  - Indirection block

File types

- Regular files
- Directories
- Hardlinks
- Symbolic links
- Block and character devices
- Sockets (Unix, raw and IP)

Permissions, attributes and ACLs

- Unix defaults to 12 bits for permissions (see chmod(1))
  - Often written in octal
  - user, group, others
  - 4 – read, 2 – write, 1 – execute
  - 3 bits are special
- Attributes extend this to make files immutable, append-only, etc (see chattr(1))
- ACLs can be used to grant access to only single users/groups

Processes

- A process is a unit of execution
  - A program is a file on disk
  - An executing program is a process
- All processes have a parent, an owner and a group
- The owner is usually the user who started the process
- Any process without an explicit parent is a child of the special process init with PID 1
- Daemons

Filehandles

- Any process which opens a file gets a numeric identifier for the file.
  - This is the filehandle
- There are three special filehandles for all processes
  - 0 – STDIN
  - 1 – STDOUT
  - 2 – STDERR

Redirecting I/O

- <
  - Redirect stdin from another file
- >
  - Redirect output to a file, overwriting
- >>
  - Redirect output, appending
- 2>
  - Redirect stderr
- 2>&1
  - Redirect stderr to stdout

Chaining commands

- Use the | operator
  - command1 | command2
  - Generically, c1 | c2 | c3 |c4 ...
- | connects the stdout of command1 to the stdin of command2
- Build long chains of commands incrementally

Shells for programming

- Unix shells are programming languages
- A shell script is a sequential list of commands
- The shebang line #!
- Loops
  - for, select, while, until
- Compound statements
  - if then [else]
  - case

Variables

- Variables are untyped
- Variable names start with $ for /bin/sh
  - echo $foo
  - echo “This string has an embedded $variable”
  - echo “This file is named ${var}txt”
- Assignment
  - foo=”bar”
  - file_list=`ls -1 /bin`
  - file_list=$(ls -1 /bin)

Environment variables

- Some variables are set on boot, or on login.
  - The totality of these variables defines the user environment
- Common examples (use “set” to see them all):
  - $PATH
  - $HOME
  - $UID
  - $IFS

Profiles

- For bourne shells, global profiles are in /etc/profile
- Personal overrides are configurable in ~/.bash_profile
  - ~ is an alias for $HOME
  - Filenames starting with a '.' are “hidden” files and often used for configuration

Job control

- End a command with & to run it in the background
- Use **jobs** to see jobs running in the background
- Use **fg** to bring a job to the foreground
- Use **Ctrl+Z** (suspend) and **bg** to send tasks to the background

Task scheduling

- For a one off task, use **atd**
- For tasks repeated at regular intervals, use the cron daemon (**crond**)
- Cron config files are found in
  - /etc/crontab
  - /etc/cron.*
  - /etc/cron.d/*
  - /var/spool/cron/* ← user crons
- Use crontab -e to edit user crons

Package management

- rpm/deb → packaging file format
- yum/dnf/apt → wrappers around rpm/deb to do dependency management and provide features like searching
- Always stick to one packaging format, provided by your OS
  - Programming languages provide their own package management
  - Do NOT use this on production

Building packages

- Use mock(8) to build RPMs
  - In desperate cases, use fpm
- Learn to write spec files
- Python, Perl and Ruby come with modules which will generate spec files for you

Packaging for Debian/Ubuntu

- Use sbuilder or sbuild
- Or use fpm

AAA

- Users are traditionally identified by a login name
- Authentication is traditionally done by a password
- Audit logging and accounting is via system logs

User information storage

- Traditionally /etc/passwd with hashed passwords in /etc/shadow
- In larger environments, LDAP, Active Directory and/or Kerberos are used.
- In certain cases, SSL certificates can be used for mutual authentication, with the username in the SSL certificate.

SSH

- Originally designed as a replacement for rsh(1)
- Encrypts data on the wire
- Supports authentication via private/public key pair
- Secure file copying (scp(1))
- Can be used to tunnel other programs
- sssd(8) can look up public keys in LDAP

SSH Keys

- ssh-keygen
- Use a good passphrase
- authorized_keys(5)
- ssh agent
- ssh-add

Privilege escalation

- root
  - The single administrative user
  - Never login or work as this user
- su
  - Switch User
  - Switch current context to other user.
  - Needs password of other user
- sudo
  - More fine-grained control of privileges
  - Does not need password of other user

Syslog

- Syslog is a standard protocol for system and network logging
- Linux has a choice of syslog daemons
  - syslogd
  - rsyslog
- All of these can log locally and remotely

Limitations of syslog

- Syslog runs over UDP on the network
  - Syslog messages can be lost
  - Syslog messages have a short message length limit
- Syslog messages get logged to files, so searching can be difficult
- Not all applications use syslog
  - Apache
  - Java

ELK

- The ELK system refers to ElasticSearch, Logstash and Kibana
- Elasticsearch is a distributed Java system for indexing (ES is a wrapper over Lucene)
- Logstash is a log shipper which will ship flat files, syslog and a number of other forms of messages
- Kibana is a Javascript application which is a frontend to ElasticSearch

Editing text

- Linux offers a lot of text editors
  - vi/vim
  - emacs
  - joe
  - pico/nano
  - ed

Which editor?

- This is a religious question
- vi is everywhere
- emacs is a very good editor/IDE for programmers
- Most Linux systems will offer vim

Notes on vi

- Original Unix editor was ed – a line editor
- vi (visual) is a modal editor
  - Command mode
    - Move around the file
  - Edit mode
    - Add/change text

A quick vi cheatsheet

- <esc> - command mode
- i – Insert before cursor
- I – Insert at beginning of line
- a – append after the cursor
- A – Append to end of line
- :w <file> - write to named file (or current file)
- :q - quit

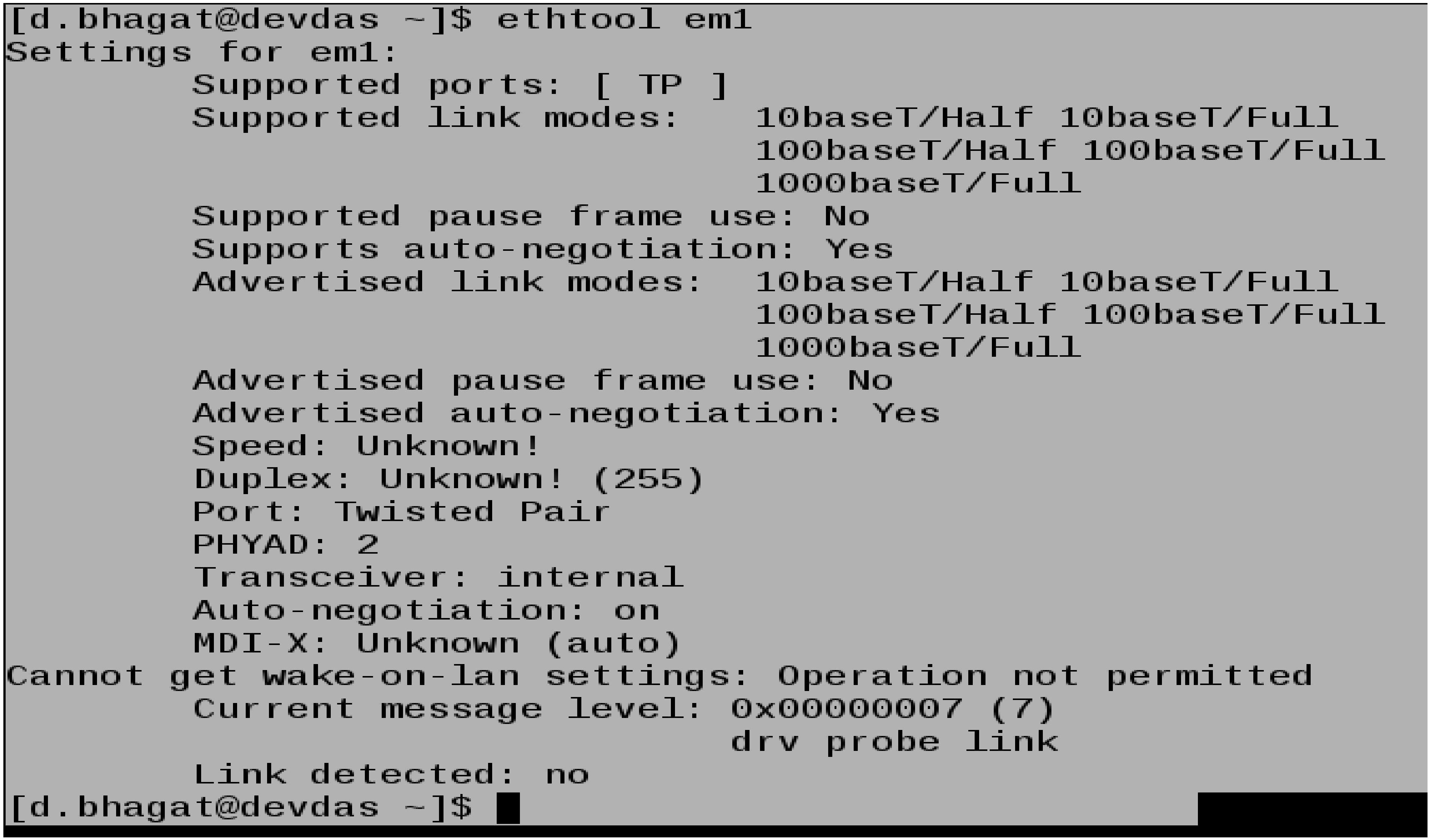
A quick vi cheatsheet

- :q! - Quit without saving
- y – Copy text
- <n> yy – Copy <n> line(s)
- yw – Copy next word
- d – delete text
- <n>dd – Delete <n> line(s)
- dw – Delete next word
- p - Paste

Networking basics

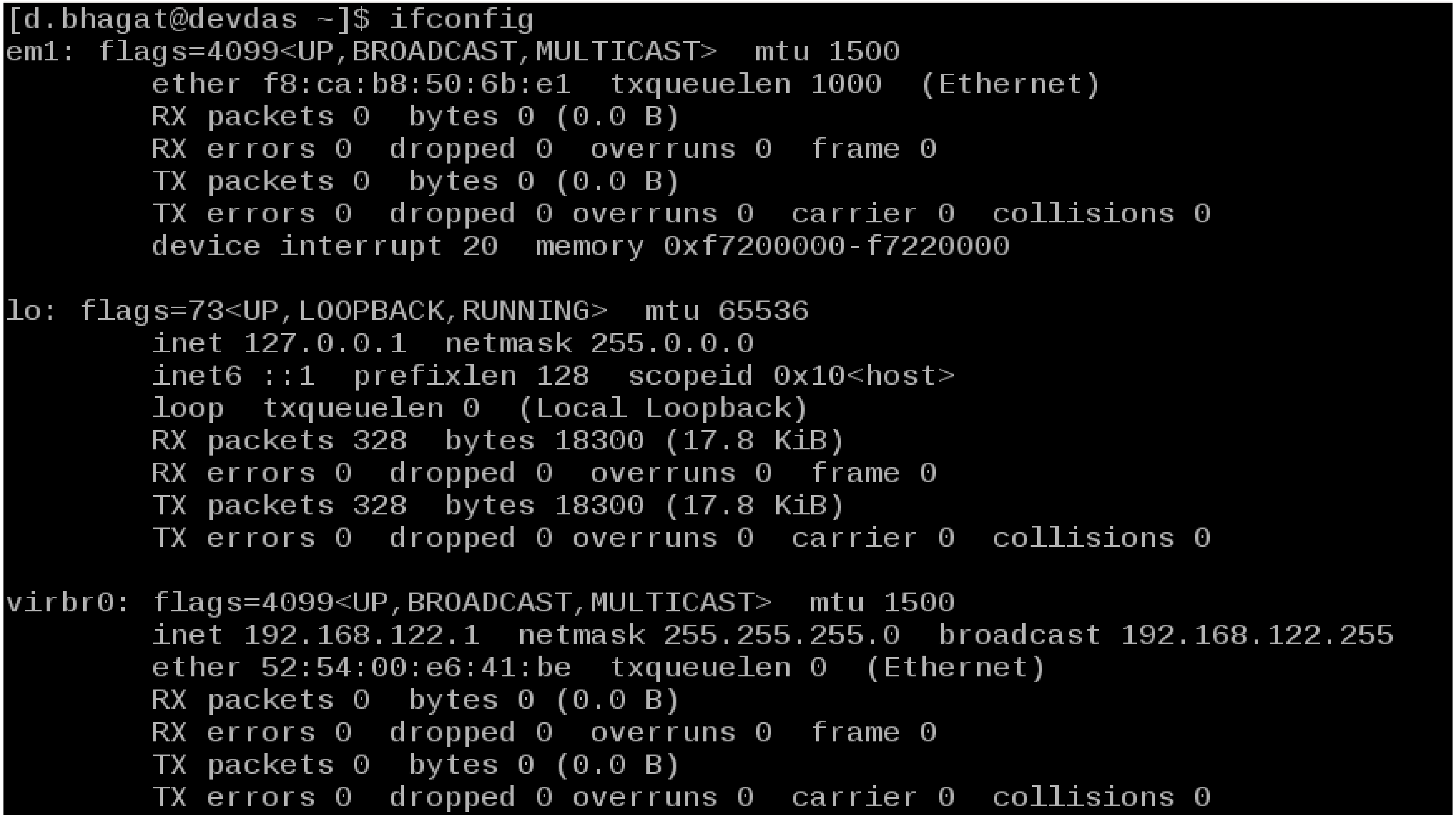
- ethtool
  - Information about ethernet interfaces
- ifconfig
  - Shows information about interfaces
- ip
  - Replacement for ifconfig and route

Examples



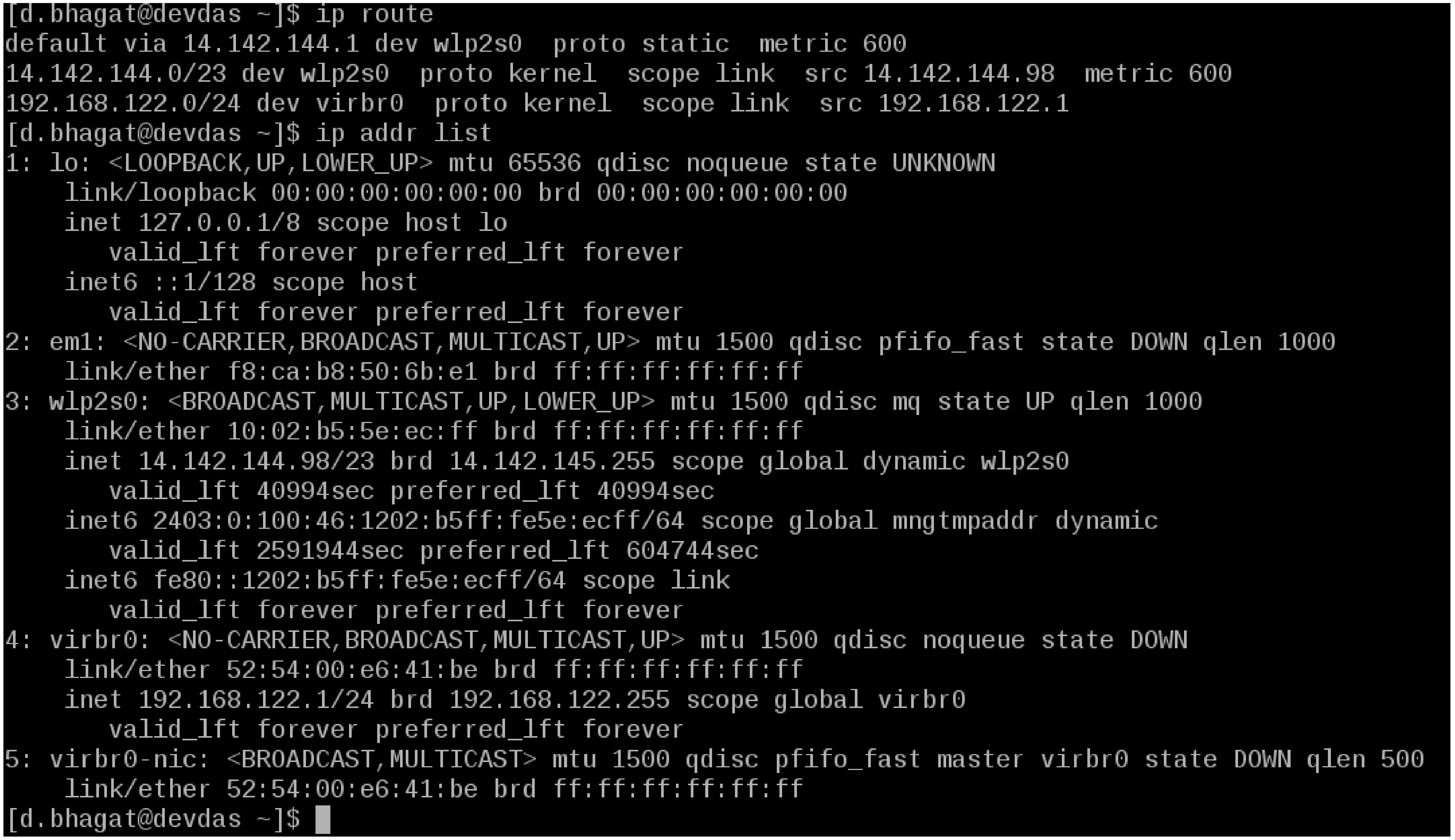
```
[d.bhagat@devdas ~]$ ethtool em1
Settings for em1:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supported pause frame use: No
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Advertised pause frame use: No
        Advertised auto-negotiation: Yes
        Speed: Unknown!
        Duplex: Unknown! (255)
        Port: Twisted Pair
        PHYAD: 2
        Transceiver: internal
        Auto-negotiation: on
        MDI-X: Unknown (auto)
Cannot get wake-on-lan settings: Operation not permitted
        Current message level: 0x00000007 (7)
                               drv probe link
        Link detected: no
[d.bhagat@devdas ~]$ █
```

Examples



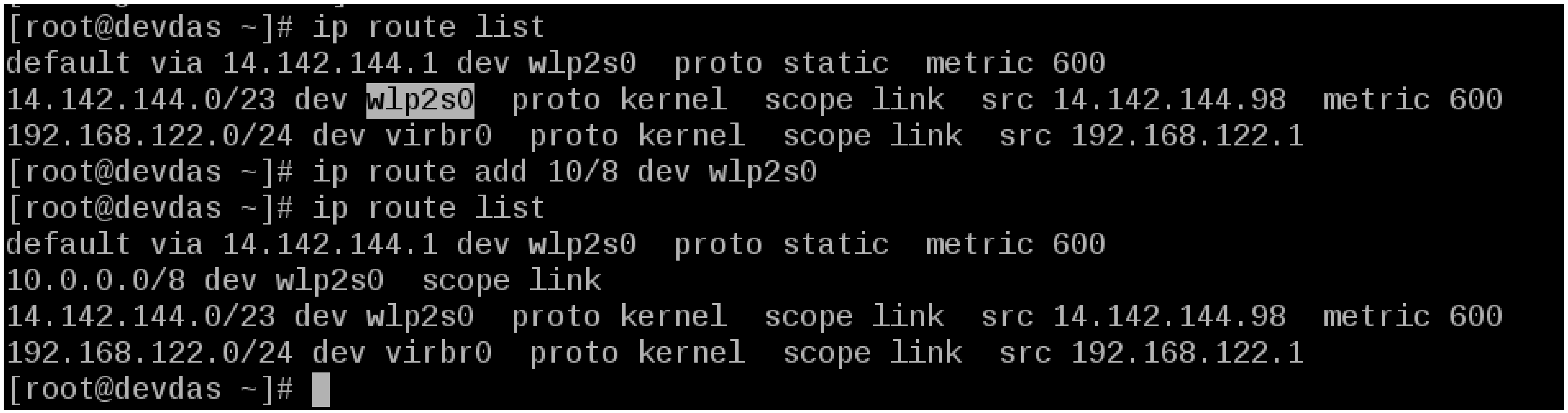
```
[d.bhagat@devdas ~]$ ifconfig
em1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether f8:ca:b8:50:6b:e1  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xf7200000-f7220000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 0  (Local Loopback)
        RX packets 328  bytes 18300 (17.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 328  bytes 18300 (17.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 52:54:00:e6:41:be  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Examples



```
[d.bhagat@devdas ~]$ ip route
default via 14.142.144.1 dev wlp2s0  proto static  metric 600
14.142.144.0/23 dev wlp2s0  proto kernel  scope link  src 14.142.144.98  metric 600
192.168.122.0/24 dev virbr0  proto kernel  scope link  src 192.168.122.1
[d.bhagat@devdas ~]$ ip addr list
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: em1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state DOWN qlen 1000
    link/ether f8:ca:b8:50:6b:e1 brd ff:ff:ff:ff:ff:ff
3: wlp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether 10:02:b5:5e:ec:ff brd ff:ff:ff:ff:ff:ff
    inet 14.142.144.98/23 brd 14.142.145.255 scope global dynamic wlp2s0
       valid_lft 40994sec preferred_lft 40994sec
    inet6 2403:0:100:46:1202:b5ff:fe5e:ecff/64 scope global mngtmpaddr dynamic
       valid_lft 2591944sec preferred_lft 604744sec
    inet6 fe80::1202:b5ff:fe5e:ecff/64 scope link
       valid_lft forever preferred_lft forever
4: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
    link/ether 52:54:00:e6:41:be brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever
5: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 state DOWN qlen 500
    link/ether 52:54:00:e6:41:be brd ff:ff:ff:ff:ff:ff
[d.bhagat@devdas ~]$ 
```

Examples



```
[root@devdas ~]# ip route list
default via 14.142.144.1 dev wlp2s0  proto static  metric 600
14.142.144.0/23 dev wlp2s0  proto kernel  scope link  src 14.142.144.98  metric 600
192.168.122.0/24 dev virbr0  proto kernel  scope link  src 192.168.122.1
[root@devdas ~]# ip route add 10/8 dev wlp2s0
[root@devdas ~]# ip route list
default via 14.142.144.1 dev wlp2s0  proto static  metric 600
10.0.0.0/8 dev wlp2s0  scope link
14.142.144.0/23 dev wlp2s0  proto kernel  scope link  src 14.142.144.98  metric 600
192.168.122.0/24 dev virbr0  proto kernel  scope link  src 192.168.122.1
[root@devdas ~]#
```

Bridging

- Linux supports bridged interfaces (particularly useful with virtual machines) – L2 packet forwarding
  - brctl addbr br0
  - brctl addif br0 em0 em1
- Use ebtables to enable packet control based on MAC address

Bonding

- You can bond interfaces on Linux to get higher throughput, redundancy, or both

```
modprobe bonding mode=0 miimon=100 # load bonding module

ifconfig eth0 down        # putting down the eth0 interface
ifconfig eth1 down        # putting down the eth1 interface

# changing the MAC address of the bond0 interface
ifconfig bond0 hw ether 00:11:22:33:44:55
# to set ethX interfaces as slave the bond0 must have an ip.
ifconfig bond0 192.168.55.55 up

ifenslave bond0 eth0   # putting the eth0 interface in the slave mod for bond0
ifenslave bond0 eth1   # putting the eth1 interface in the slave mod for bond0
```

Bonding modes

- mode=0 (Balance Round Robin)
  - RR per packet
- mode=1 (Active backup)
- mode=2 (Balance XOR)
  - Packets to the same host go out of the same interface
- mode=3 (Broadcast)
  - Transmit on both
- mode=4 (802.3ad/LACP)
- mode=5 (Balance TLB)
  - Outgoing traffic balanced across interfaces based on interface speed
- mode=6 (Balance ALB)
  - Balances both incoming and outgoing traffic

Pushing packets

- Between interfaces
  - Set the sysctl net.ipv4.ip_forward = 1
- NAT
- Use iptables

Dynamic routing

- Use Quagga or BIRD
- Quagga consists of multiple individual daemons
  - One for each protocol
  - Always enable the zebra daemon
    - Only zebra can assign IP addresses and static routing
- Config files
  - /etc/quagga/daemons
  - /etc/quagga/${protocol}.conf
- Quagga supports Cisco like commands and syntax
- Use vtysh to get a single interface to all daemons

OSPF

```
kmint(config)# router ospf
kmint(config-router)# router-id 5.5.5.5
kmint(config-router)# passive-interface default
kmint(config-router)# no passive-interface eth0
kmint(config-router)# no passive-interface eth1
kmint(config-router)# no passive-interface lo
kmint(config-router)# network 10.1.1.2/24 area 1
kmint(config-router)# network 10.1.2.2/24 area 1
kmint(config-router)# network 5.5.5.5/32 area 1
kmint(config-router)# end
kmint#
```

BGP example

```
!
! Zebra configuration saved from vty
!   2006/06/09 16:13:05
!
hostname bgp.example.com
password zebra
enable password zebra
log file /var/log/quagga/bgpd.log
!
router bgp 65270
 bgp router-id 192.168.27.1
 network 192.168.27.0/24
 network 192.168.254.128/30
 neighbor 192.168.254.9 remote-as 65000
 neighbor 192.168.254.130 remote-as 65120
 distance bgp 150 150 150
!
line vty
```

Firewalling and packet mangling

- Linux offers excellent packet filtering capabilities using iptables
- A **rule** is a statement which expresses a decision about a packet
- A **chain** is an ordered list of rules
- A **table** is a set of chains

Examples

- /sbin/iptables -t FILTER -A FORWARD -p tcp --dport 23 -j DROP
- /sbin/iptables -t FILTER -A INPUT -p tcp --dport 53 -j ACCEPT
- /sbin/iptables -t FILTER -A INPUT -p udp -m multiport --dports 53,123 -j ACCEPT

iptables modules

- See iptables-extensions(8)
  - addrtype
  - cluster
  - comment
  - connbytes
  - connlabel
  - conntrack
  - ...

Traffic Control

```
[d.bhagat@devdas ~]$ sudo tc
Usage: tc [ OPTIONS ] OBJECT { COMMAND | help }
       tc [-force] [-OK] -batch filename
where  OBJECT := { qdisc | class | filter | action | monitor }
       OPTIONS := { -s[tatistics] | -d[etails] | -r[aw] | -p[retty] | -b[atch] [filename] |
-n[etns] name }
```

- tc controls egress traffic
- You can control bandwidth per end node
- You can give preference to latency sensitive traffic

tc

- A qdisc is a system for controlling traffic
- It is possible to configure simple, classless qdiscs or classful ones which let you classify traffic
- tc can be used to limit buffer bloat, and ensure that interactive traffic is not harmed by large batch uploads/downloads

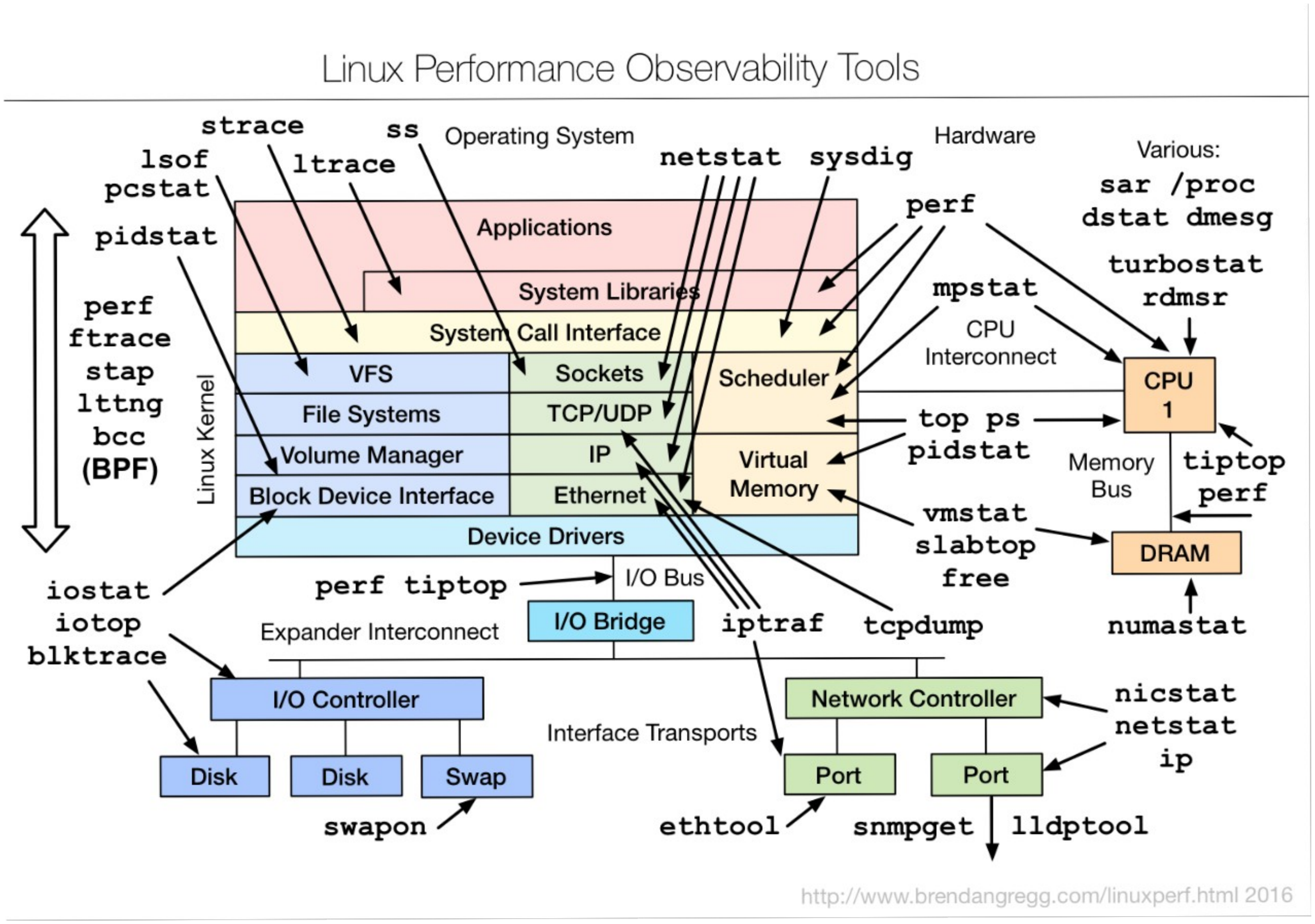
tc examples

```
# This line sets a HTB qdisc on the root of eth0, and it specifies that the class
# 1:30 is used by default. It sets the name of the root as 1:, for future references.
tc qdisc add dev eth0 root handle 1: htb default 30

# This creates a class called 1:1, which is direct descendant of root (the parent is
# 1:), this class gets assigned also an HTB qdisc, and then it sets a max rate of
#  6mbits, with a burst of 15k
tc class add dev eth0 parent 1: classid 1:1 htb rate 6mbit burst 15k

# The previous class has this branches:

# Class 1:10, which has a rate of 5mbit
tc class add dev eth0 parent 1:1 classid 1:10 htb rate 5mbit burst 15k

# Class 1:20, which has a rate of 3mbit
tc class add dev eth0 parent 1:1 classid 1:20 htb rate 3mbit ceil 6mbit burst 15k

# Class 1:30, which has a rate of 1kbit. This one is the default class.
tc class add dev eth0 parent 1:1 classid 1:30 htb rate 1kbit ceil 6mbit burst 15k
```

Intrusion Detection Systems

- Host based
  - Tripwire
  - AIDE
  - OSSEC
- Network based
  - SNORT

Monitoring and debugging



More tools

- traceroute
- tracepath
- mtr
- lsof
- smokeping
- wireshark
- netcat
- Nagios/Icinga/Opsview/Check_mk
- Riemann
- SNMP

Devops

- A culture of developers, operations, network engineers, DBAs, security …, all working together to improve business.
- CAMS
  - Culture
  - Automation
  - Measurement
  - Sharing

Configuration Management

- Hosts should be sheep, not pets
- Configuration management software is expected to make hosts with the same role alike
  - Puppet
  - Chef
  - Salt
  - Cfengine
- Orchestration software makes sequential changes on different hosts
  - Ansible

CfgMgmt

- Promises
  - A system will get to a given state
  - Trust based on past performance
  - Policy based management
- Modeling
  - Humans should not express the process to get to a given state. They should just say what the final state needs to be.

CMDB

- The CMDB is the single source of truth for your entire IT system
- Configuration management **MUST** derive information from the CMDB
- The CMDB does not express policy

Change management

- Make small, frequent changes
  - Large changes carry higher risks
  - Make rollbacks easier
- Use a version control tool
  - git, bzr, hg, svn, …
- Code reviews
  - gerrit

Writing good commit messages

Header line: explain the commit in one line (use the imperative)

Body of commit message is a few lines of text, explaining things
in more detail, possibly giving some background about the issue
being fixed, etc etc.

The body of the commit message can be several paragraphs, and
please do proper word-wrap and keep columns shorter than about
74 characters or so. That way "git log" will show things
nicely even when it's indented.

Make sure you explain your solution and why you're doing what you're
doing, as opposed to describing what you're doing. Reviewers and your
future self can read the patch, but might not understand why a
particular solution was implemented.

Advanced topics

- Programming
  - Perl, Python, Ruby
- Databases
  - PostgreSQL, MySQL
- Security
- DNS
  - BIND, PowerDNS, Knot, DNSSEC, DNSCurve, dnsdist
- Email
  - Postfix, Sendmail, Exim, Courier, Cyrus IMAP, Dovecot, spam...
- Web
  - Apache, nginx, varnish, squid, ...
- Distributed filesystems
  - NFS, CEPH, GlusterFS, …
- Key-Value stores
  - Consul, Kubernetes

References

- http://shop.oreilly.com/product/9780596154493.do
- http://shop.oreilly.com/product/9781593273897.do
- http://shop.oreilly.com/product/9781565922259.do
- http://books.cat-v.org/computer-science/unix-programming-environment/
- http://books.cat-v.org/computer-science/unix-programming-environment/
- https://www.amazon.com/Unix-Network-Programming-Vol-Networking/dp/B000OIAXBY
- https://www.amazon.com/UNIX-Network-Programming-Interprocess-Communications/dp/0132974290
- https://www.amazon.com/TCP-Illustrated-Protocols-Addison-Wesley-Professional/dp/0321336313
- https://www.amazon.com/TCP-IP-Illustrated-Implementation-Vol/dp/020163354X

References

- https://www.amazon.com/Design-UNIX-Operating-System/dp/0132017997
- https://www.amazon.com/UNIX-Programming-Addison-Wesley-Professional-Computng/dp/0131429019
- https://www.amazon.com/Elements-Programming-Style-2nd/dp/0070342075
- http://www.opsschool.org/
- https://www.fprintf.net/vimCheatSheet.html
- http://vim.rtorr.com/
- http://www.brendangregg.com/linuxperf.html

References

- www.lartc.org/lartc.html
- https://wiki.quagga.net/wiki/index.php/ConfigurationExamples
- https://cyruslab.net/2012/05/12/configuration-examples-with-quagga/
- http://bird.network.cz/?get_doc&f=bird-3.html
- https://wiki.archlinux.org/index.php/Advanced_traffic_control
- http://www.linuxhorizon.ro/bonding.html
- https://wiki.debian.org/BridgeNetworkConnections
- https://blog.chef.io/2010/07/16/what-devops-means-to-me/

References

- http://riemann.io/
- https://www.nagios.org/
- http://graphite.readthedocs.io/en/latest/
- http://grafana.org/
- https://www.elastic.co/
- https://www.digitalocean.com/community/tutorials/an-introduction-to-selinux-on-centos-7-part-1-basic-concepts
- https://wiki.centos.org/HowTos/SELinux

References

- http://chris.beams.io/posts/git-commit/
- http://tbaggery.com/2008/04/19/a-note-about-git-commit-messages.html
- https://www.git-scm.com/book/en/v2/Distributed-Git-Contributing-to-a-Project#Commit-Guidelines
- https://github.com/torvalds/subsurface/blob/master/README#L82-109
- https://puppet.com/
- https://www.chef.io/
- https://saltstack.com/
- https://www.ansible.com/

References

- https://git-scm.com/
- https://www.mercurial-scm.org/
- http://bazaar.canonical.com/en/
- https://www.gerritcodereview.com/
- http://shop.oreilly.com/product/0636920039846.do
- https://artofmonitoring.com
- https://snort.org/
- https://nmap.org/
- http://nc110.sourceforge.net/